JOURNAL OF COMPUTATIONAL PHYSICS 67, 37–58 (1986)

A Block Preconditioned Conjugate Gradient-type Iterative Solver for Linear Systems in Thermal Reservoir Simulation

SRINIVAS BETTÉ

Mobil Research and Development Corporation, Dallas Research Laboratory, P. O. Box 819047, Dallas, Texas 75381-9047

JULIO C. DIAZ

Parallel Processing Institute, University of Oklahoma, 202 W. Boyd, Suite 219, Norman, Oklahoma 73019

WILLIAM R. JINES

Mobil Research and Development Corporation, P. O. Box 819047, Dallas, Texas 75381-9047

AND

TROND STEIHAUG

Statoil, Stavanger, Norway

Received May 6, 1985; revised November 27, 1985

A preconditioned residual-norm-reducing iterative solver is described. Based on a truncated form of the generalized-conjugate-gradient method for nonsymmetric systems of linear equations, the iterative scheme is very effective for linear systems generated in reservoir simulation of thermal oil recovery processes. As a consequence of employing an adaptive implicit finite-difference scheme to solve the model equations, the number of variables per cell-block varies dynamically over the grid. The data structure allows for 5- and 9-point operators in the areal model, 5-point in the cross-sectional model, and 7- and 11-point operators in the three-dimensional model. Block-diagonal-scaling of the linear system, done prior to iteration, is found to have a significant effect on the rate of convergence. Block-incomplete-LU-decomposition (BILU) and block-symmetric-Gauss–Seidel (BSGS) methods, which result in no fill-in, are used as preconditioning procedures. A full factorization is done on the well terms, and the cells are ordered in a manner which minimizes the fill-in in the well-column due to this factorization. The convergence criterion for the linear (inner) iteration is linked to that of the nonlinear (Newton) iteration, thereby enhancing the efficiency of the computation. The algorithm, with both BILU and BSGS preconditioners, is evaluated in the context of a variety of thermal simulation problems. The solver is robust and can be used with little or no user intervention. © 1986 Academic Press, Inc.

0021-9991/86 $3.00

Copyright © 1986 by Academic Press, Inc.
All rights of reproduction in any form reserved.

I. INTRODUCTION

Oil reservoir models are based on mass and energy balances, and some auxiliary relations, forming a set of coupled, time-dependent, nonlinear partial differential equations. Most reservoir simulators in the petroleum industry employ a finite-difference formulation to solve these equations. A partially implicit or a fully-implicit (THERMS [1]) discretization leads to a set of nonlinear algebraic equations to be solved every time-step. Newton's method is commonly used to linearize these equations, generating systems of sparse linear equations. One of the most time-intensive steps of the simulation is the solution of these linear systems. Simulation of field-size problems require the use of large numbers of grid blocks. Solution of large sparse linear systems using direct elimination is impractical because it is both memory and work intensive. Hence the need for an efficient iterative method. In this paper we will be concerned with linear systems generated in thermal simulation in general, and simulator THERMS [1] in particular.

Thermal problems, in general, are highly nonlinear because of heat effects contributed by phase changes and combustion reactions. The Jacobian matrix generated in these problems is block structured, strongly nonsymmetric, and non-diagonally dominant. Due to the heterogeneity of the reservoirs, the coefficients of the equations and consequently the matrix elements vary significantly, sometimes by several orders of magnitude. Methods such as LSOR are often found to diverge or converge very slowly. Also, in these methods, parameter estimation is very difficult because the linear systems change continually. These changes are significant especially when wells are turned on/off. Our objective is to develop an algorithm which is not only efficient but also robust in the sense that it can be applied to a wide variety of thermal oil recovery problems with little or no user intervention.

We present an iterative method which is an extension of the conjugate-gradient-method first introduced by Hestenes and Stiefel [2]. Recently there have been several developments in the application of conjugate gradient-type iterative methods to these types of linear systems [3, 4, 5, 6, 7, 8, 9, 10]. These iterative methods have the characteristic that hard to find parameter estimation is not required. The rate of convergence can be accelerated by orthogonalization and by preconditioning. In this algorithm, preconditioning takes two forms: prior to iteration, a block-diagonal scaling is done, and during iteration an incomplete factorization of the scaled system is used to adjust the direction of the iteration. Block-incomplete-LU-decomposition (BILU) and block-symmetric-Gauss–Seidel (BSGS) incomplete factorizations are considered as preconditioning procedures during iteration. However, a full factorization is carried out on the well terms. The fill-in in the well column resulting from this factorization is minimized using a special cell ordering. The overall efficiency of the simulator depends upon the stopping criterion for the linear (inner) iteration and the termination rule for the nonlinear (outer) iteration [11]. A procedure linking these criteria enhances the efficiency of the computation.

To evaluate the performance of the algorithm, it is applied to various thermal

simulation problems. Steam and combustion processes are considered in the areal, cross sectional, and 3D modes. Here the objective is to determine the relative performances of the two preconditioners, and to identify deficiencies, if any, for any particular type of process or model.

In Section II we briefly describe the thermal model, its nonlinear finite-difference form, and the generation of linear systems. In the following section we introduce the residual-norm-reducing method leading upto the generalized-conjugate-residual method. Sections IV and V describe the preconditioned-conjugate-residual-method and the two preconditioners considered in this work. The decomposition of the bordered well equations is presented in Section VI. In Section VII we consider different cell ordering to study its effect on fill-in in the well wings during decomposition of the well equations. The convergence criterion for the linear solver and its relation to the nonlinear residual is presented in Section VIII. Section IX contains the description of the test problems. The results are presented in Section XI.

II. Linear Systems in THERMS [1]

The model (THERMS) developed to simulate thermal oil recovery processes (Hwang, Jines, and Odeh [1]), consists of an energy balance equation, overall mass balance equations for the three phases (oil, water, and gas), and component mass balances for steam, oxygen, and inert gas. The general form of the balance equations is

Energy balance:

$$\frac{\partial}{\partial t}\left[(1-\phi_f)\,\rho_f C_f T + \phi_f(h_o\rho_o S_o + h_w\rho_w S_w + h_g\rho_g S_g - \gamma P)\right]$$

$$= k_f\nabla^2 T - \nabla\cdot(h_o B_o + h_w B_w + h_g B_g) + (\Delta H)R + Q_H.$$

Overall phase mass balance equations:

$$\frac{\partial}{\partial t}(\phi_f\rho_k S_k) = -\nabla\cdot B_k - \alpha_k R + Q_k,$$

$k = o, w, g$ for oil, water, and gas phase.

Gas phase component mass balance equations:

$$\frac{\partial}{\partial t}(\phi_f\rho_g S_g y_i) = -\nabla\cdot(y_i B_g) + v_i R_w + \bar{\alpha}_i R + Q_i,$$

$i = s, O_2, n$ for steam, oxygen, and inert.

In addition, there are some auxiliary relationships and nonlinear well constraints; they are

(i) The phase velocities are characterized by Darcy's law:

$$B_k = \frac{k\, k_{rk} \rho_k}{\mu_k} \nabla \Psi_k; \qquad \Psi_k = P - \rho_k^w z \cdot \mathbf{g},$$

(ii) $S_o + S_w + S_g = 1$,

(iii) $y_s + y_{O_2} + y_n = 1$,

(iv) equations of state for ρ_g, ρ_o, ρ_w, and ϕ_f,

(v) phase equilibrium relationship for steam: $y_s = f(P, T)$ or steam tables,

(vi) viscosity, relative permeability, and enthalpy relationships for all phases,

(vii) initial condition and boundary conditions ($k = 0$ at the boundaries),

(viii) well constraints:

$$Q_k = F(P - P_w); \qquad F = f(P, T, S_k),$$

Q_k is fixed for rate controlled wells (P_w is an unknown) P_w is fixed for pressure controlled wells.

This is a set of coupled, nonlinear, second-order, time-dependent partial differential equations in three space dimensions. Hwang *et al.* [1] showed that this system of differential equations can be reduced to a set of partial differential equations corresponding to the primary unknowns P, T, S_o, and S_w (P, T, S_o, S_w and y_{0_2} if the moving-front algorithm is not used). Moreover, the primary unknowns can change depending on the state of the system; for example, if $S_w = 0$, then the variable S_w is replaced by y_s: steam fraction. For a more detailed description of the model, the reader is referred to [1].

The backward Euler scheme is used for time discretization and the implicit central-difference method, combined with one point upstream weighting is used for space discretization. This implicit finite-difference discretization results in a set of nonlinear difference equations which are to be solved for the primary unknowns at each time level for every grid block (node/cell) simultaneously. Let the nonlinear system for each grid block be

$$f_j = f(v_j) = 0, \qquad j = 1, 2, \ldots, \tag{1}$$

where v_j is the vector of unknowns for the jth block; for example, $v_j = (P, T, S_o, S_w)_j$. In THERMS the equations are collected on a cell-by-cell basis, which gives a cell block structure to the system of equations. The complete system can be written as

$$F(u) = \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_n \end{bmatrix} = 0, \tag{2}$$

where $u = (v_1, v_2, ..., v_n)$ is the vector of all unknowns. Newton's method is used to linearize this system. For a given approximation $u = u_0$, successive approximations for u are determined using

$$u^{k+1} = u^k - J^{-1} F(u^k), \tag{3}$$

where J is the Jacobian matrix: $J_{ij} = \partial F_i / \partial u_j$.

Rewriting (3), we have

$$J(u^{k+1} - u^k) = -F(u^k),$$

i.e.,

$$J \Delta u = -F(u^k). \tag{4}$$

This is a linear system of algebraic equations with unknowns Δu and has the form

$$Ax = b. \tag{5}$$

The Jacobian matrix A is sparse, and is generally nonsymmetric and nondiagonally dominant. Furthermore, A has a natural block structure. For example, consider a grid system with natural ordering, as shown in Fig. 1. For this grid system the figure shows the structure of the Jacobian matrix for a 9-point operator. Each entry in the matrix is a block corresponding to the unknowns at the cell. Moreover, since the number and the type of unknowns depend on the physical state of the system, they can vary dynamically from cell to cell. In THERMS, the well constraints and the cell flow equations are coupled. Consequently, the nonlinear system of equations takes the form

$$\begin{bmatrix} F(u) \\ F_w(u_w) \end{bmatrix} = 0 \tag{6}$$

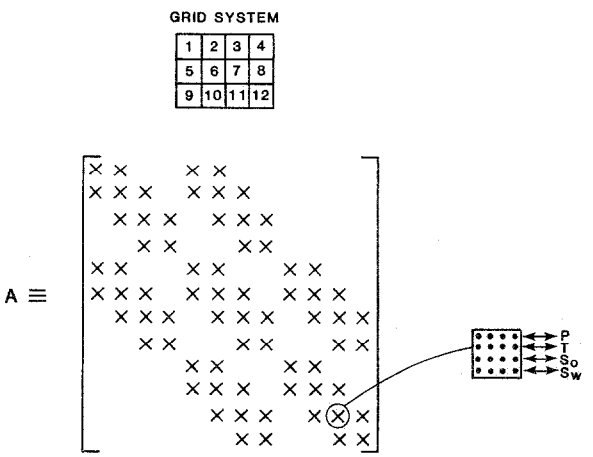

FIG. 1. Structure of the Jacobian matrix.

which in turn gives rise to a bordered linear system of the form

$$\left[\begin{array}{c|c} A & V \\ \hline H & W \end{array}\right] \begin{bmatrix} x_c \\ x_w \end{bmatrix} = \begin{bmatrix} b_c \\ b_w \end{bmatrix}, \tag{7}$$

where A contains derivatives of flow equations with respect to cell unknowns, V corresponds to derivatives of flow equations with respect to well unknowns, H corresponds to derivatives of the well equations with respect to cell unknowns, and W corresponds to derivatives of the well equations with respect to well unknowns. x_c and x_w are the cell and well unknowns, respectively. Generally the number of grid blocks far exceeds the number of well constraints, resulting in small well-borders in the linear system. Since the number of cell-completions for each well is small compared to the total number of grid blocks, the well-wings V and H are sparse. The test problems reflect typical numbers that one encounters in thermal simulation. A linear system of this type is to be solved at every Newton iteration. Cell ordering and its effect on the structure of the matrix will be discussed later in conjunction with preconditioners.

III. Residual-Norm-Reducing Methods

Consider a linear system of equations,

$$Ax = b,$$

where A is a nonsingular matrix, b is a given vector, and x is the unknown solution vector to be determined. Given an approximate initial guess for the solution vector x_1, these methods determine recursively the next approximation vector $x_{j+1}, j = 1, 2, 3,....$ Residual-norm-reducing methods minimize $\|r_{j+1}\|_2$ the l_2-norm of the residual vector

$$r_{j+1} = b - Ax_{j+1}, \tag{8}$$

along a search direction d_j. The method determines a parameter λ_j for which

$$x_{j+1} = x_j + \lambda_j d_j. \tag{9}$$

Several choices for generating the direction could be considered [3, 5, 12]. In this work the direction vector is based on the latest residual.

From (8) and (9), it follows that

$$r_{j+1} = r_j - \lambda_j A d_j.$$

To minimize the l_2-norm of r_{j+1}, one can show that it suffices to require that

$$(r_{j+1}, Ad_j) = 0,$$

thus

$$\lambda_j = \frac{(r_j, Ad_j)}{(Ad_j, Ad_j)}. \tag{10}$$

For nonsymmetric problems, like those generated by THERMS, it is useful to require orthogonality of the current direction with respect to the previous directions [5, 13]. The orthogonality condition is that d_{j+1} be A-orthogonal to the previous directions $d_1,..., d_j$. That is,

$$(Ad_{j+1}, Ad_i) = 0 \qquad (i = 1, 2,..., j).$$

Let $\bar{d} = r_j$ be the direction considered, and define d_{j+1} as

$$d_{j+1} = \bar{d} - \sum_{i=1}^{j} \beta_i d_i. \tag{11}$$

By enforcing the A-orthogonality condition on d_{j+1} with respect to $d_1,..., d_j$, we obtain

$$\beta_i = \frac{(A\bar{d}, Ad_i)}{(Ad_i, Ad_i)}. \tag{12}$$

Since it is not practical to store all previous directions, only a number, say K, of them is kept. Hence, (11) can be rewritten as

$$d_{j+1} = \bar{d} - \sum_{i=j-\Theta(j,K)+1}^{j} \beta_i d_i. \tag{13}$$

If we choose $\Theta(j, K) = \min(j, K)$ then the method corresponds to ORTHOMIN(K) [5, 13]. On the other hand, if we choose to restart the orthogonalization process after every K iterations then $\Theta = j \bmod(K)$, and the method corresponds to the generalized-conjugate-residual (GCR) method with K orthogonalizations, GCR(K) [5]. The GCR(K) method requires the same memory as ORTHOMIN(K) but requires less work per step. In this work we have used the GCR(K) method.

IV. Preconditioning Iterative Methods

The basic idea of preconditioning consists of modifying the original system in such a way that the iteration on the modified system can proceed at a faster rate. Thus, preconditioning acts to accelerate the iterative process. In general, preconditioning consists of replacing the system

$$Ax = b,$$

by an equivalent system

$$Cy = d,$$

where C and d are derived by matrix pre- and/or post-multiplication. More specifically, let Q be a square-invertible matrix. The preconditioned systems can take on one of the following forms:

$$\begin{aligned} Q^{-1}Ax &= Q^{-1}b \qquad \text{(left preconditioning)},\\ Q_{\mathrm{L}}^{-1}AQ_{\mathrm{R}}^{-1}Q_{\mathrm{R}}x &= Q_{\mathrm{L}}^{-1}b \qquad \text{(split preconditioning)},\\ AQ^{-1}Qx &= b \qquad \text{(right preconditioning)}. \end{aligned}$$

Depending on sparsity considerations, preconditioning can be performed prior to iteration or during the iterative process. In this algorithm, a block-diagonal scaling of the equations is performed prior to iteration [14]. Since this scaling acts upon the equations (range space), it is called a left preconditioning. Thus, the residual that is minimized is a scaled residual. However, during the iterative process, this algorithm uses an incomplete block-decomposition of the scaled matrix (defined in the following section). This incomplete decomposition is applied from the right and is performed at the generation of the direction. It in effect scales the solution (domain space). The preconditioned-residual-norm-reducing method is obtained by replacing A by AQ^{-1} and x by Qx in Eqs. (8)–(13). This is equivalent to replacing and redefining the directional vector $d_j = Q^{-1}d_j$. The algorithm is summarized in Appendix A.

V. Preconditionings

In THERMS all the information of the fluid-flow and energy equations is gathered at the cell level. Further, since the information is ordered by cell number, the matrices generated have a natural block structure. THERMS supports a variety of discretizations: 5- and 9-point operators in the areal mode, 5-point in the cross-sectional mode, and 7- and 11-point in the three-dimensional mode. The model also allows for a number of nonneighbor connections. These options are made transparent to the linear solver via a sparse data structure similar to Yale Sparse Pak [15], using row major ordering in the block entries of A and column major in the entries of individual blocks. Let

$$A = L + D + U,$$

where D is the block-diagonal of the matrix A, L, and U are strictly lower and upper block triangular, respectively.

The block-diagonal scaling replaces this matrix with the matrix

$$C \equiv D^{-1}A = D^{-1}L + I + D^{-1}U.$$

An incomplete block-decomposition of the matrix A has the form

$$Q = (L + D_n)\, D_n^{-1} (U + D_n),$$

where D_n is a block-diagonal matrix. Block-incomplete-LU-decomposition requires D_n to satisfy

$$D_n = D - \text{block diagonal}[LD_n^{-1}U].$$

The block-symmetric-Gauss–Seidel requires

$$D_n = D.$$

The BILU decomposition has the property that the block diagonal of $(A - Q) \equiv 0$. This incomplete decomposition is a block version of the DKR decomposition [16, 17, 19].

The effect of block-diagonal scaling is shown in Fig. 2, where the square of the l_2-norm of the pressure equation residuals is plotted against the number of linear

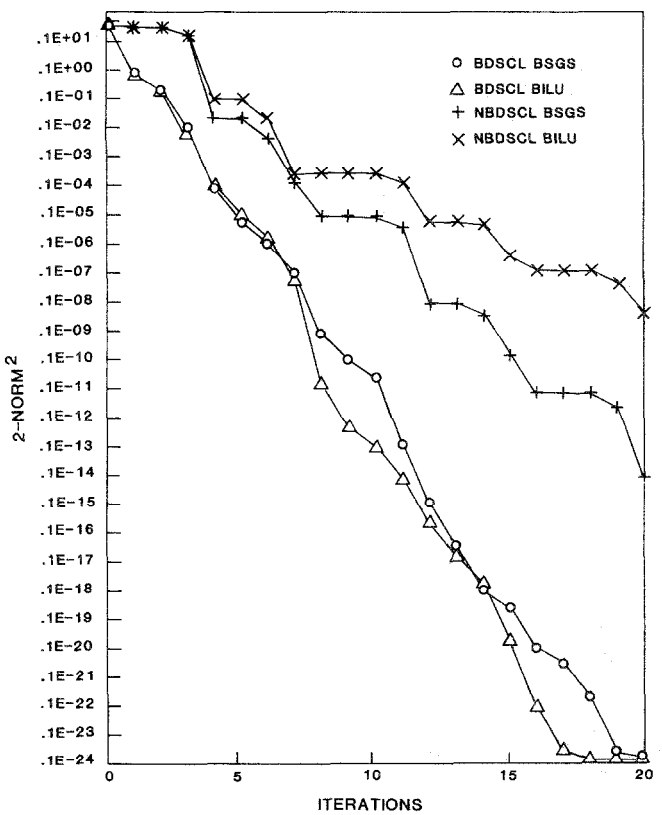

FIG. 2. Effect of block-diagonal scaling on rate of convergence (pressure errors).

iterations for two different preconditioners (BILU and BSGS). It is easy to see that the iterative process converges faster with block-diagonal scaling (BDSCL) than without scaling (NBDSCL). A more detailed analysis of the effects of scaling can be found in Diaz *et al.* [14]. It should be mentioned that in many cases, there was no convergence without scaling. Hence, block-diagonal scaling seems essential for linear systems in THERMS.

VI. Well Equations Decomposition

Assume that an incomplete block decomposition of $A(=L+D+U)$ is available. Then, an incomplete factorization of the bordered matrix of system (7) would have the form

$$\left[\begin{array}{c|c} L+D_n & 0 \\ \hline H_f & L_w \end{array}\right] \times \left[\begin{array}{c|c} D_n & 0 \\ \hline 0 & I \end{array}\right]^{-1} \times \left[\begin{array}{c|c} U+D_n & V_f \\ \hline 0 & U_w \end{array}\right]. \tag{14}$$

This decomposition procedure consists of applying Gaussian elimination to all the well terms. The complete factorization of the well terms is feasible because of the small number of well constraints. The factorization yields

$$\begin{aligned} V_f &= D_n(L+D_n)^{-1}V, \\ H_f &= H(U+D_n)^{-1}D_n, \\ L_w U_w &= W - H_f D_n^{-1} V_f. \end{aligned} \tag{15}$$

This incomplete decomposition ((14) and (15)) works well in terms of the number of iterations. It has been observed that the number of iterations required to solve the bordered system (7) is never greater than the number of iterations required to solve A by itself [9, 19]. The well constraints are satisfied exactly.

VII. Cell Ordering

The densely coupled incomplete factorization (14) and (15) could lead to some fill-in V_f, H_f and $L_w U_w$. Consider a two-well system in a simple cross-sectional model as shown in Fig. 3a. A natural order of the cells results in a matrix structure with complete fill-in in the well wings (Fig. 3b). A simple reordering of the cells, with cells having well completions (of multicompleted wells) ordered last, results in an elimination of the fill-in (Fig. 3c). However, this type of cell ordering does not completely eliminate fill-in in general, but it does reduce the extent of fill-in as shown in Figs. 4a and b. It should be pointed out that because THERMS makes

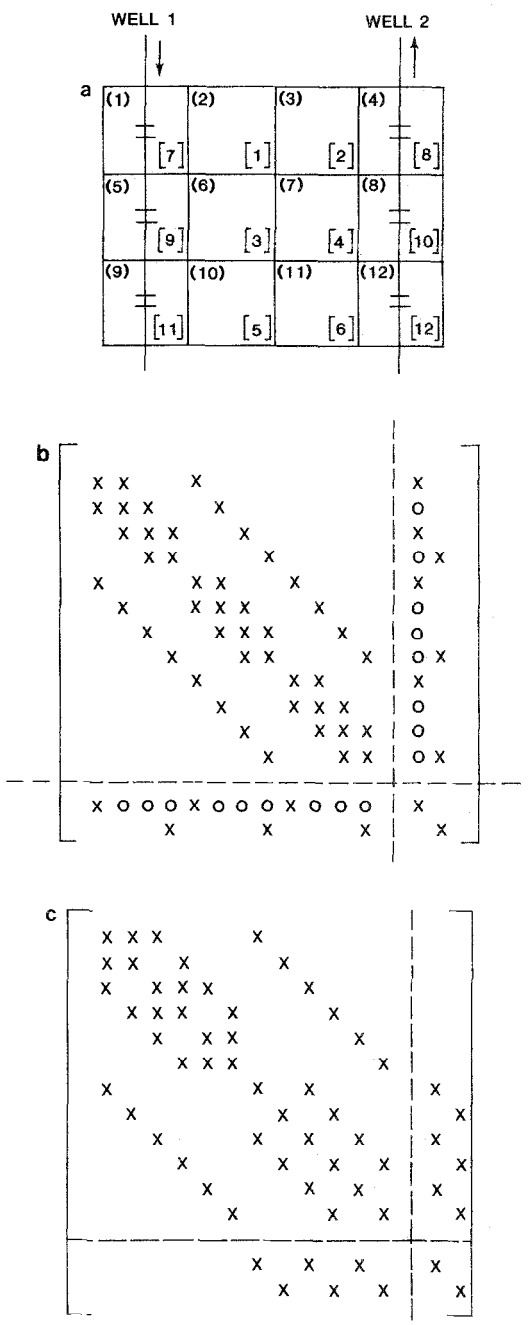

FIG. 3 (a) Grid system: ( ) natural order of reservoir cells, [ ] multicompleted well cells ordered last; (b) matrix structure for natural order of cells, (x) entries, (o) fill-in; (c) matrix structure for multicompleted well cells ordered last.

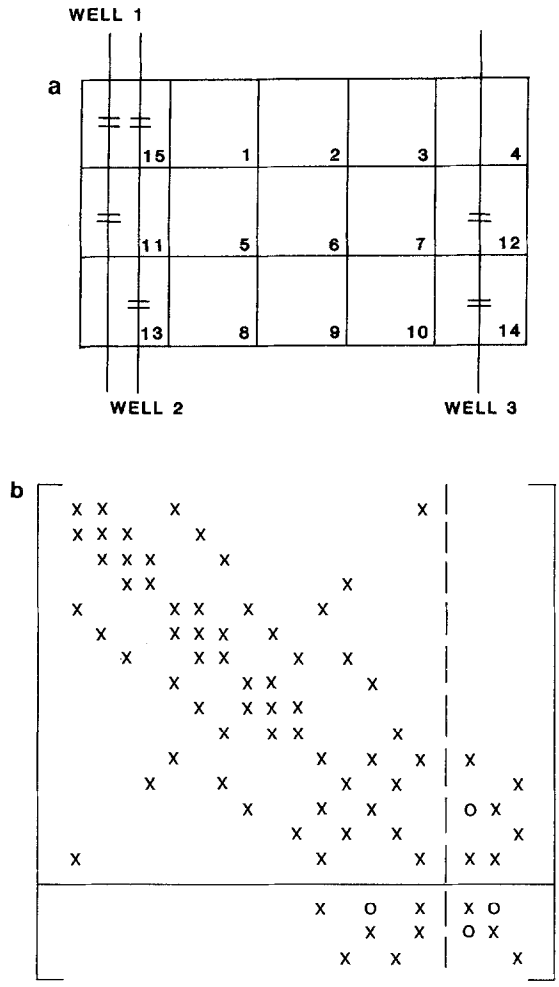

FIG. 4. (a) Grid system with multicompleted well cells ordered last; (b) matrix structure for grid system in Fig. 4a.

extensive use of indirect addressing, the different orderings are handled with no additional difficulty. Also, since the well equations are factored completely, this reordering of cells should not affect the performance adversely. All test problems presented here use cell ordering of the type shown in Fig. 4a.

VIII. CONVERGENCE CRITERION

The convergence of the iterative process is measured in various forms: one form is to require the l_2-norm of the residual be reduced below a certain preset tolerance,

is injected for 10 days at capacity with constraints of maximum bottom hole pressure of 1000 psia and maximum rate of 1000 STB (CWE)/day. After a seven-day soak period following injection, the well goes on production for 348 days, thus completing a cycle. The production constraints are minimum bottom hole pressure of 17 psia and a maximum production rate of 1000 STB/day.

Problem 5 considers steam drive in an inverted nine-spot pattern with severe well constraints. A $\frac{1}{2}$ element of symmetry is simulated using a $9 \times 9 \times 5$ grid system. Steam is injected at capacity with constraints: maximum bottom hole pressure of 1000 psia at the center of top layer and maximum injection rate of 300 STB (CWE)/day. The producers (both far and near) produce at capacity with the following constraints: minimum bottom hole pressure of 17 psia at the center of top layer, maximum production rate of 1000 STB/day of liquids and maximum steam rate of 10 STB (CWE)/day. All other data is as given in Table II. The objective of this case is to study the performance of the solver when the wells reach capacity and change modes. A total of 2000 days was simulated.

Problem 6 is a field-scale simulation of an in-situ combustion process including the effects of a strong aquifer. The field consists of a single O_2 injection well and 27 production wells. The field is represented by 5000 nodes (100×50 areal grid system) of which 961 are fully implicit and the rest are, implicit pressure, semi-implicit temperature and explicit saturations. The formation is 31% porous, 35-ft thick, 2700-ft deep, at 1200 psia, 125°F, 35% oil saturated and 63% water saturated. The oil has a density of 57.4 lbs/ft^3, and viscosity of 227 cp at 100°F, 3.37 cp at 300°F, and 0.47 cp at 600°F. The fuel concentration in the system is 0.0109 lb of oil/lb of rock. The rate controlled injector injects O_2 at 240 M SCF/day and at 446°F.

X. Results and Discussion

The simulation of the processes described by the above problems was carried out for long periods of time so that one could determine the average performance of the solver. The average values of the number of Newton iterations per time-step, the number of linear iterations per Newton iteration and the fraction of overall time spent solving linear equations are summarized in Table III, for both preconditioning procedures. The ratios of the linear equation solution cpu time per Newton iteration, the total linear equation solution time and the overall model cpu time, of the two preconditioning procedures are also tabulated in Table III. Since our objective is to study the relative performance of the preconditioners, absolute cpu time which is a function of the machine used, degree of vectorization, programming structure, etc., is often misleading and is not presented.

In general, the solver with BILU performed very well. In some instances, when the time-step was large the solver failed to converge to a specified tolerance within the preset maximum number of iterations. An automatic time-step reduction feature rectified this behavior in all cases, and the simulation progressed. In Table III we

TABLE III

Problem Number	Number of blocks	BILU				BSGS				Ratio of linear system time per Newton itr. (BSGS/BILU)	Ratio of total lin. system time (BSGS/BILU)	Overall model cpu time ratio (BSGS/BILU
		Number of time steps	Newton iterations per time step	Linear itr. per Newton itr.	Fraction of overall time spent solving lin. equations	Number of time steps	Newton itr. per time step	Linear itr. per Newton itr.	Fraction of overall time spent solving lin equations			
1	1024	157	3.17	5.21	0.57	158	3.63	5.6	0.49	0.61	0.7	0.83
2	245	1167	5.53	2.6	0.59	1088	5.53	2.82	0.43	0.54	0.51	0.68
3	245	526	3.77	4.85	0.59	796	4.85	3.41	0.43	0.46	0.89	1.22
4	52	252	2.4	5.72	0.63	164	2.91	3.19	0.5	0.4	0.32	.4
5	324	152	3.52	3.57	0.65	166	3.26	4.05	0.43	0.59	0.6	.81
6	5000	460	2.89	4.11	0.53	435	2.88	4.35	0.45	0.76	0.72	.85

observe that on an average the problems required 2.5 to 6 Newton-steps per time step and 2.5 to 6 linear iterations per Newton iteration. With BILU approximately 53-65% of the total cpu time was spent solving the linear systems.

BSGS leads to a slightly higher number of Newtons per time step and linear iterations per Newton iteration compared to the BILU preconditioner. However, the work involved in constructing BSGS is significantly lower. This is reflected in the fraction of time spent solving linear equations, which is 10-15% lower than BILU. The difference between the two preconditioners due to vectorization is expected to be small because of short vectors (blocks are 4×4 or less) and indirect addressing. No effort was made either to vectorize specific segments or to suppress vectorization. We expect similar results on a scalar machine. The ratio of the linear system time per Newton iteration and total linear system time shows that BSGS is faster by a factor of 1.3 to 2.5 and 1.1 to 3.1, respectively. However, the ultimate test of the procedure is the overall speed of simulation. We stress the fact that in this

time-step selection strategy. Since the iterative solver yields solutions only to a certain accuracy in a global sense, BILU and BSGS result in a slightly different solution path. Moreover, the error criterion in the nonlinear and linear steps are linked. Consequently, the number of time-steps and the number of Newton steps are different for the two procedures used. We note that in Problem 3 (3D in-situ combustion) simulation with BSGS takes more total time than with BILU, even though the linear equation time per Newton iteration is lower by a factor of two. In this case, to simulate the same number of days, a higher number of time steps were required with BSGS than with BILU. The same behavior was observed in cross-sectional models of in-situ combustion processes. But in the areal model BSGS out performs BILU. Indeed this observation is problem-dependent and hence cannot be generalized.

In certain cases when the time step size was allowed to grow very large, we observed that the solver with BSGS preconditioner reduced the l_2-norm of the residual to a preset tolerance, but, produced an inaccurate solution vector, eventually leading to a run off of the simulation producing physically unreasonable results. By limiting the maximum time-step size, we were able to avoid this behavior. The solver with BILU, on the other hand, did not exhibit this behavior. Further, it was observed that without diagonal scaling the solver with BSGS performed poorly. Hence, in concert with block-diagonal scaling, BSGS seems to perform reasonably well in most cases, making it a good first choice. However, for "hard" combustion problems BILU may be a better choice.

To study the performance of the solver at and around breakthrough, Problem 1 was run for 1000 days (past breakthrough) with a steam injection rate of 800 STB (CWE)/day, using the BILU preconditioner. The solver performed well while showing only a slight increase in the average number of iterations past breakthrough. In Problem 2, the injector reached capacity resulting in a change in the mode of operation of the well. However, no problems relating to the solver were encountered. In Problem 5, severe constraints imposed on the production wells led to

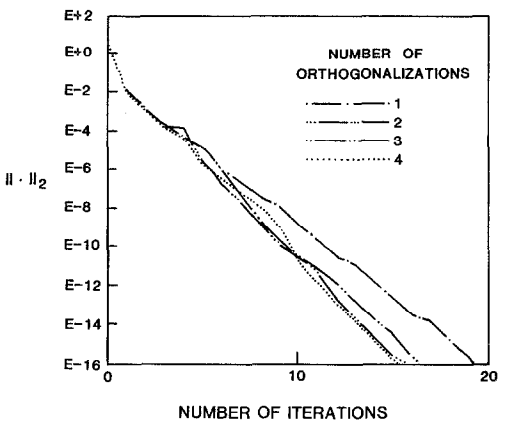

FIG. 5. Effect of number of orthogonalization vectors on rate of convergence.

changes in well operation from being pressure controlled to being rate controlled. This sudden change, and in addition, having to solve more equations did not negatively affect the solver performance. Problem 6 demonstrates the performance of the solver for large-field-size problems. In this combustion example both preconditioners performed well with BSGS preconditioner being more efficient in terms of overall cpu time.

Thus far, we have not considered the effect of the orthogonalization parameter, K. During the iterative process, fewer orthogonalizations means lower work requirements. Some studies [7, 18] indicate that 10–15 orthogonalizations may be necessary. In all the calculations presented here the maximum number of orthogonalizations used is 3. A test on the rate of convergence as a function of the number of orthogonalizations was conducted using Problem 2. In Fig. 5 the rate of convergence for 2, 3, and 4 orthogonalizations is essentially the same, but the rate of convergence with one orthogonalization is slower. So, for this algorithm, 2 orthogonalizations seem to be adequate.

XI. CONCLUSIONS

We have developed an iterative linear equation solver based on the generalized-conjugate-gradient method. The robustness of the algorithm is demonstrated by applying it to various test cases derived from oil recovery processes using thermal methods. The BILU preconditioner performs very well for all the cases considered. The BSGS preconditioner, which is less work intensive than BILU, performs reasonably well for most cases and hence is a good first choice. BILU preconditioner seems to be a better choice for tough combustion problems. For this algorithm, two orthogonalization vectors seem to be adequate.

APPENDIX A

Algorithm Preconditioned Residual-Norm-Reducing Method with "K" Orthogonalizations

Data	A	Nonsingular (Jacobian) matrix,
	b	specified vector (nonlinear residuals),
	ϕ	stopping parameter,
	M	maximum number of iterations permitted.
	K	number of orthogonalizations.
	Q	preconditioning (incomplete decomposition) of A
Putput	x	approximate solution vector,
	r	accumulated residual vector satisfying $\|r\|_2 < \phi \|b\|_2$
Internal parameters	$d_1, d_2, ..., d_k$	minimization direction vector,
	λ	stepsize,
	j	pointer for direction,
	θ	number of directions in memory,
	β_n	projection over previous direction,
	σ_j	l_2-norm of Ad_j squared.

BEGIN:
$x = 0$
$r = b$,
$d_1 = Q^{-1} r$
$j = 1$,
$\theta = 1$,
$\sigma_j = (Ad_j, Ad_j)$
loop on i, from 1 until M, each 1,
$\lambda = (r, Ad_j)/\sigma_j$
$x = x + \lambda d_j$
$r = r - \lambda A d_j$
if ($\|r\|_2 < \phi \|b\|_2$) then exit loop,
else:
$j = 1 + j \bmod(K)$,
$\Theta = \min(\Theta + 1, K)$, (or $\Theta = j$)
$\bar{d} = Q^{-1} r$;

loop on m, from 2 until Θ each 1
$n = 1 + (j + K - m) \bmod(K)$
$\beta_n = (A\bar{d}, Ad_n)/\sigma_n$
$d_j = \bar{d} - \beta_n d_n$
$Ad_j = A\bar{d} - \beta_n Ad_n$
end loop on m.
$\sigma_j = (Ad_j, Ad_j)$
end loop on i,
END.

APPENDIX B: NOMENCLATURE

A	Jacobian matrix
b	nonlinear residual
b_c	nonlinear residual of flow equations
b_w	nonlinear residual of well equations
b_0	initial nonlinear residual vector for time-step
B_k	molar flux defined by Darcy's law for kth phase $k = o, w, g$ (mols/ft^2/d)
C	scaled or preconditioned Jacobian matrix
C_f	specific heat of the formation (Btu/lb-°R)
C_o	compressibility of oil (1/psi)
Cp_o	heat capacity of oil (Btu/lb-°R)
CWE	cold water equivalent
d_j	minimization direction of iterative method
D	block diagonal of block diagonal splitting of A
D_n	block diagonal defining incomplete block decomposition
g	gravity vector
h_k	specific enthalpy of the kth phase, $k = o, w, g$(Btu/lb mol-°R) wells term row
H_f	wells term row after fill-in during decomposition
J	Jacobian matrix
k	permeability of the formation (darcies)
k_{rk}	relative permeability of the kth phase, $k = O, w, g$
k_f	thermal conductivity of the formation (Btu/ft-d-°R)
K	number of orthogonalizations
L	strictly lower block triangular of block diagonal splitting of A
L_w	lower triangular factor of filled-in diagonal well term
Mw_o	molecular weight of oil
P	pressure (psi)
P_w	well bottom hole pressure (psi)
Q	preconditioning matrix
Q_H	heat source (sink) due to wells (Btu/ft^3-d)
Q_i	mass source (sink) due to wells of ith component in the gas phase, $i = s, O_2, n$(lb mols/ft^3 – d)
Q_k	mass source (sink) due to wells of kth phase, $k = o, w, g$ (lb mol/ft^3 – d)
r_j	linear iteration residual

S_g	gas saturation
S_{gi}	initial gas saturation
S_o	oil saturation
S_{oi}	initial oil saturation
S_w	water saturation
S_{wi}	initial water saturation
STB	stock tank barrel
T	temperature (°R)
u	vector of all unknowns
u_0	initial guess for u
U	strictly upper block triangular of block diagonal splitting of A
U_w	upper triangular factor of filled-in diagonal term
v_j	vector of unknowns for the jth block
V	wells term column
V_f	wells term column after fill-in during elimination
W	wells term diagonal
x	vector of unknows in linear system or (Δu)
x_c	vector containing reservoir cell variable changes
x_w	vector containing well pressure changes
x_j	iterative approximation solution vector
y_i	mole fraction of the ith component in the gas phase, $i = s, O_2, n$,
z	depth (ft)
α_k	combustion reaction stoichiometric coefficient for the kth phase
$\bar{\alpha}_i$	combustion reaction stoichiometric coefficient for the ith component in the gas phase
β_j	orthogonalization coefficient
γ	unit conversion factor for converting mechanical energy to heat
Γ	parameter describing linear convergence rate of the nonlinear (outer) iteration
δ	coefficient of thermal expansion ($°R^{-1}$)
ΔH	heat of combustion reaction (Btu/lb mol)
$\Theta(j, K)$	truncation formula for orthogonalization of direction of iterative method
λ_j	conjugate-gradient minimization step
ν_i	stoichiometric coefficient of water vaporization–condensation: $\nu_s = 1$, $\nu_{O_2} = \nu_n = 0$
μ_k	viscosity of the kth phase (cp)
ρ_k	molar density of the kth phase (lb mol/ft^3)
ρ_k^w	mass density of the kth phase (lb/ft^3)
σ_j	l_2-norm of the product of the Jacobian matrix A and d_j
ϕ	stopping criterion parameter
ϕ_f	porosity of the formation
ψ_k	flow potential for the kth phase
Ω	parameter describing quadratic convergence rate of the nonlinear (outer) iteration
$\|z\|_2$	l_2-norm of vector, $\equiv (z, z)^{1/2}$
(z, y)	inner product of vectors z and y

ACKNOWLEDGMENTS

The authors would like to acknowledge the helpful discussions with A. H. Dogru, M. S. El-Mandouh, and M. B. Ray. We are especially grateful to A. E. McDonald for his help and advice. We would like to thank Dr. A. S. Odeh for his encouragement in this work and C. C. Hendrick for her skillful typing of the manuscript.

REFERENCES

1. M. K. HWANG, W. R. JINES, AND A. S. ODEH, *Soc. Pet. Eng. J.* April, 271 (1982).
2. M. R. HESTENES AND E. STIEFEL, *J. Res. Natl. Bur. Stand.* **49**, 409 (1952).
3. R. CHANDRA, "Conjugate Gradient Methods for Partial Differential Equations," Yale University, New Haven, CT, Research Report No. 129, January 1979.
4. O. AXXELSSON, *Linear Algebra Appl.* **29**, (1980).
5. H. ELMAN, "Iterative Methods for Large, Sparse, Nonsymmetric Systems of Linear Equations," Yale University, New Haven, CT, Research Report No. 229, April 1982.
6. J. R. WALLIS AND K. H. COATS, SPE 10491, VI Symposium on Reservoir Simulation, New Orleans, LA, February 1982.
7. J. R. WALLIS, SPE12265, VII Symposium on Reservoir Simulation, San Francisco, CA, November 1983.
8. J. A. MEYERINK, SPE 12262, VII Reservoir Simulation Symposium, San Francisco, CA, November 1983.
9. A. BEHIE AND P. A. FORSYTH, JR. SPE 12263, VII Reservoir Simulation Symposium, San Francisco, CA, November 1983.
10. P. CONCUS, G. H. GOLUB, AND D. P. O'LEARY, "A Generalized Conjugate Gradient Method for the Numerical Solution of Elliptic Partial Differential Equations," *Sparse Matrix Computations*, Bunch, J. R., and Rose, D. J., Eds. (Academic Press, New York 1976), pp. 309–332.
11. J. C. DIAZ, W. R. JINES, AND T. STEIHAUG, SPE 13500, VIII Reservoir Simulation Symposium, Dallas, TX, February 1985.
12. D. M. YOUNG AND N. C. JEA, *Linear Algebra Appl.* **34**, 159 (1980).
13. P. K. W. VINSOME, SPE 5729, IV Symposium on Reservoir Simulation, Los Angeles, CA, 1976.
14. J. C. DIAZ, W.R. JINES, A. E. MCDONALD, AND T. STEIHAUG, *Commun Appl. Numer. Methods*, in press.
15. S. C. EISENSTAT, M. C. GURSKY, M. H. SCHULTZ, AND A. H. SHERMAN, *Int. J. Numer. Methods Eng.* **18**, 1145 (1982).
16. T. DUPONT, R. P. KENDALL, AND H. H. RACHFORD, JR., *SIAM J. Numer. Anal.* **5**, 559 (1968).
17. A. BEHIE AND P. K. VINSOME, SPE 9303, 55th Annual Fall Technical Conference, SPE, Dallas, TX, September 21–24, 1980.
18. A. BEHIE AND P. A. FORSYTH, JR., *SIAM J. Sci. Stat. Comput.* **5**, 543 (1981).
19. R. H. TRIMBLE AND A. E. MCDONALD, *Soc. Pet. Eng. J.* **21**, 454 (1981).